武汉大学计算机学院

本科生实验报告

**LWP音乐系统**

专 业 名 称 ：计算机科学与技术

课 程 名 称 ：大型应用软件设计

指 导 教 师 ：马于涛

组长 ：陈柏钧

小组成员 ：陈惟逸

郑宗汕

冯珞轩

陈昊阳

张钦博

文柯力

二○二一年十一月

**郑 重 声 明**

本人呈交的实验报告，是在老师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本实验报告的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本实验报告的知识产权归属于培养单位。

摘 要

本次大型应用软件课程设计实验中，我们小组开发一个web端的音乐播放网站LWP音乐系统，使用户能够检索并播放需求的音乐，随时分享自己的听歌感受，创建属于自己的歌单，用户的听歌喜好能被网站进行分析并推荐音乐，为用户创造一个良好的听歌平台。

本次实验我们小组使用了vue框架、css、html语言以及golang语言实现。本文档包括项目需求分析、系统设计说明、项目实现说明、测试报告。

**关键词：** LWP音乐系统；vue框架；golang

目录

目录

[一.项目概述 6](#_Toc87704490)

[1.1 项目代码库 6](#_Toc87704491)

[1.2 团队分工 6](#_Toc87704492)

[二.需求规格说明书 7](#_Toc87704493)

[2.1 引言 7](#_Toc87704494)

[2.1.1 编写目的 7](#_Toc87704495)

[2.1.2 项目目标 7](#_Toc87704496)

[2.1.3 系统概述 7](#_Toc87704497)

[2.1.3 项目有关人员 8](#_Toc87704498)

[2.1.4项目实现环境与技术支持 8](#_Toc87704499)

[2.2系统需求概述 8](#_Toc87704500)

[2.2.1 用例图 8](#_Toc87704501)

[2.2.2 用例的概要描述 9](#_Toc87704502)

[2.2.3 假设和依赖 10](#_Toc87704503)

[2.3 功能需求 10](#_Toc87704504)

[2.3.1 LWP音乐系统功能 10](#_Toc87704505)

[2.4 非功能需求 11](#_Toc87704506)

[2.4.1 安全性 11](#_Toc87704507)

[2.4.2 易用性 12](#_Toc87704508)

[2.4.3 可靠性 13](#_Toc87704509)

[2.4.4 性能 14](#_Toc87704510)

[2.5. 运行环境规定 14](#_Toc87704511)

[2.6. 验收验证标准 14](#_Toc87704512)

[2.6.1 界面验收标准 15](#_Toc87704513)

[2.6.2 功能验收标准 16](#_Toc87704514)

[三.系统设计说明书 18](#_Toc87704515)

[3.1 总体设计 18](#_Toc87704516)

[3.1.1 系统业务关系图 18](#_Toc87704517)

[3．1.2 系统后端请求处理流程图 18](#_Toc87704518)

[3.2 系统模块设计 19](#_Toc87704519)

[3.3 功能模块设计 19](#_Toc87704520)

[3.3.1 ui模块 19](#_Toc87704521)

[3.3.2 搜索模块 20](#_Toc87704522)

[四.项目实现 21](#_Toc87704523)

[4.1 数据准备 21](#_Toc87704524)

[4.1.1 数据来源说明 21](#_Toc87704525)

[4.1.2 系统后端接口调用参数说明记录 21](#_Toc87704526)

[4.2 前端实现 28](#_Toc87704527)

[4.2.1 Vue.js介绍 28](#_Toc87704528)

[4.2.2 前端项目结构 29](#_Toc87704529)

[4.2.3 router设置 30](#_Toc87704530)

[4.3 后端实现 32](#_Toc87704531)

[4.3.1 逻辑实现 32](#_Toc87704532)

[4.3.2 后端构建 33](#_Toc87704533)

[五.测试报告 38](#_Toc87704534)

[5.1 引言 38](#_Toc87704535)

[5.1.1 编写目的 38](#_Toc87704536)

[5.1.2 项目背景 38](#_Toc87704537)

[5.1.3 系统简介 38](#_Toc87704538)

[5.2 测试任务概述 39](#_Toc87704539)

[5.2.1 目标 39](#_Toc87704540)

[5.2.2 运行环境 39](#_Toc87704541)

[5.3 测试演示 40](#_Toc87704542)

[5.3.1 lwp音乐系统主页效果测试 40](#_Toc87704543)

[5.3.2登录窗口界面效果测试 41](#_Toc87704544)

[5.3.3 用户界面效果测试 42](#_Toc87704545)

[5.3.4歌曲播放测试 43](#_Toc87704546)

[5.3.5 歌词与评论界面效果测试 44](#_Toc87704547)

[5.3.6 视频/mv播放效果测试 45](#_Toc87704548)

[六、团队成员在项目中的角色与贡献 46](#_Toc87704549)

[评语： 48](#_Toc87704550)

[评分： 48](#_Toc87704551)

[评阅人： 48](#_Toc87704552)

**一.项目概述**

随着科学技术的发展，计算机科学技术日渐成熟，其强大的功能与便利性已被人们所熟知与接受。在人类社会的各个领域中，计算机科学技术发挥着越来越重要的作用。个人用户对于听音乐的需求也越来越大，用户对音乐检索与音乐播放需求越来越大。

​故我们力求与开发一个LWP音乐系统，能够使得用户需要时检索并播放需求的音乐，随时分享自己的听歌感受，创建属于自己的歌单，网能够根据用户的听歌喜好分析并推荐音乐。为用户创造一个良好的听歌平台。

## 1.1 项目代码库

项目代码库： https://github.com/xueyikun/LWP

## 1.2 团队分工

- UI&UE 设计 陈柏钧 陈昊阳

- PM&User 冯珞轩 文柯力

- FE负责人 陈惟逸

- FE开发 陈惟逸 陈柏钧 陈昊阳

- Server负责人 郑宗汕

- Server开发 郑宗汕 文柯力 张钦博

- QA 张钦博 冯珞轩

# 二.需求规格说明书

## 2.1 引言

### 2.1.1 编写目的

编写本文档是为了保证项目团队按时按质地完成项目开发目标，便于项目团队成员更好地了解项目开发情况，使本软件开发工作的各个过程合理有序。以文件化的形式，将软件开发过程中各项工作的人员、分工、系统资源条件、软件开发进度计划、经费安排等问题记录下来，作为项目团队及项目关系人之间的约定与共识，团队开发和检查项目工作的依据，确保软件开发计划的开展以及软件开发的顺利进行。

### 2.1.2 项目目标

系统开发目标：

1. 系统应具有实用性、可靠性和适用性。

2.  系统能够满足用户需求，使用户访问系统并注册账号后，能够实现验证帐号登陆系统的功能。

3. 需对各个数据库进行动态管理，防止混乱。

4. 能够按照用户的条件进行数据的简单查询以及复合查询。

5. 系统能够根据用户的条件，对数据进行相应的处理。

6.  能够对用户查询的结果进行分类汇总和美观的输出。

7. 注意数据的安全性，具备数据的备份以及恢复功能。

8. 方便用户的操作，用户界面需设计的合理美观，使用户操作流畅。

### 2.1.3 系统概述

本系统面向的用户为需要检索相关音乐、播放音乐、创建自己喜好的音乐歌单的人。

本系统旨在实现以下功能：

1. 用户能注册和登陆系统。

2. 用户能够搜索到歌曲/专辑的信息

3. 系统能够为用户推荐歌曲/专辑

4. 用户可以在正在播放的音乐下面留言评论、点赞、收藏、下载

5. 用户能够查看自己创建的歌单

7. 用户可以看到播放歌曲的歌词与专辑图片

8. 用户可以在歌曲下进行评论

9. 用户可以查看到歌曲下他人的评论

### 2.1.3 项目有关人员

项目开发组成员：

- 组长：陈柏钧

- 组员：陈惟逸、郑宗汕、冯珞轩、陈昊阳、张钦博、文柯力

### 2.1.4项目实现环境与技术支持

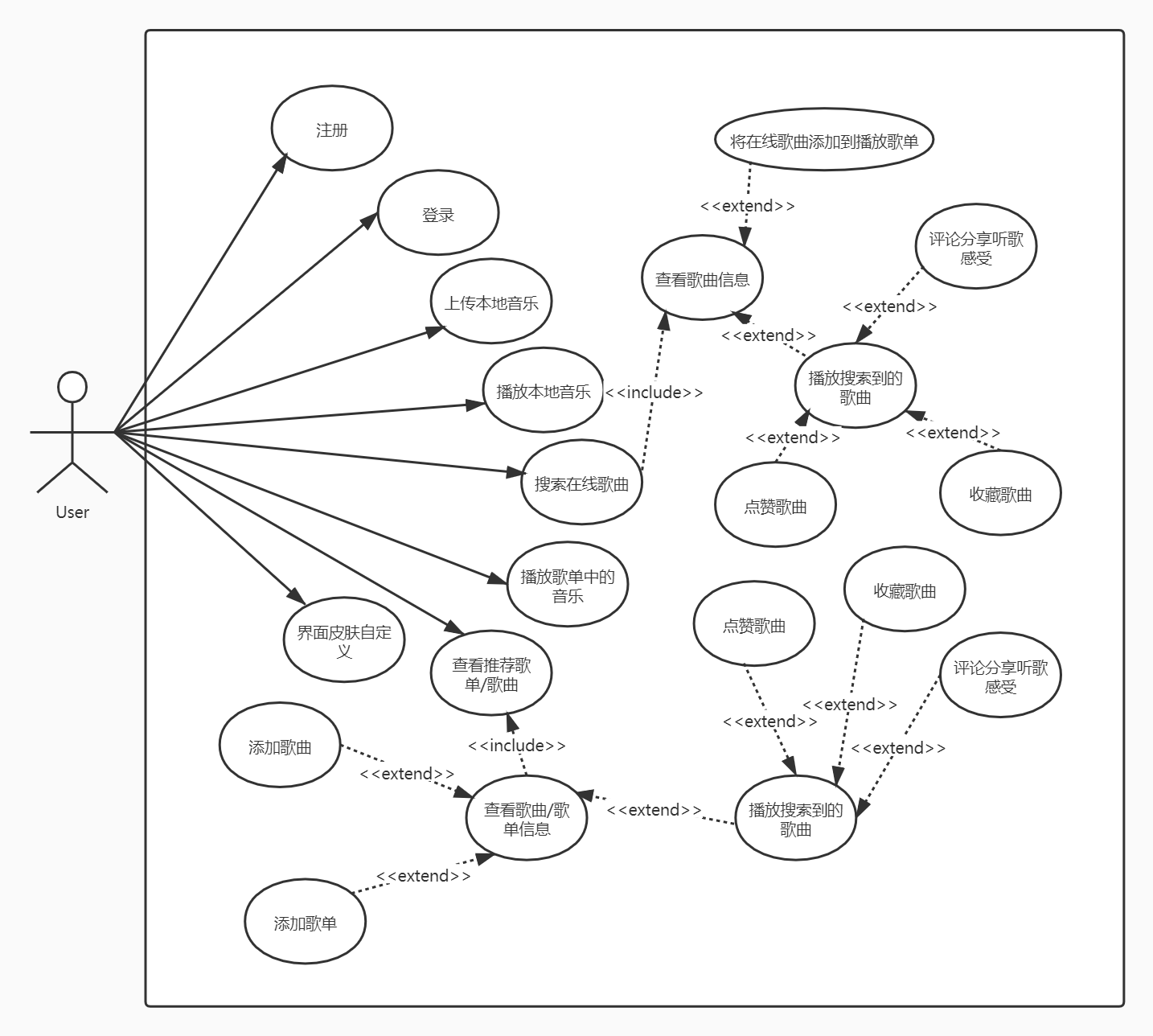
软件运行操作系统：Windows 7/8/10

开发语言：JavaScript + go

技术栈：Vue + vuex + vue-router + vue-axios + css + html

## 2.2系统需求概述

### 2.2.1 用例图



### 2.2.2 用例的概要描述

| **主要参与者** | **优先级** | **用例名** | **用例概述** |
| --- | --- | --- | --- |
| **用户** | **高** | **用户选择** | **用户通过系统注册账号，若无误则录入，否则提示重新输入.** |
| **用户** | **高** | **用户注册** | **用户通过系统登录账号，若无误则登录成功，否则提示重新输入.** |
| **用户** | **高** | **用户登录** | **执行相对应用.** |
| **用户** | **中** | **用户选择上传** | **执行相对应用.** |
| **用户** | **高** | **用户播放歌单音乐** | **执行相对应用.** |
| **用户** | **低** | **更换皮肤** | **执行相对应用.** |
| **用户** | **高** | **查看推荐歌曲** | **执行相对应用.** |
| **用户** | **高** | **用户将在线歌曲添加到歌单** | **用户在“搜索歌曲”中搜索成功后，可将在线歌曲添加到歌.** |
| **用户** | **高** | **用户播放/点赞/收藏/评论在线歌曲** | **用户在“搜索歌曲”中搜索成功后，可播放/点赞/收藏/评论该在线歌曲.** |
| **用户** | **高** | **用户播放/点赞/收藏/评论推荐歌曲/歌单** | **用户在“查看推荐歌单/歌曲”中推荐成功后，可播放/点赞/收藏/评论该在线歌曲.** |
| **用户** | **中** | **用户将推荐歌曲/歌单添加到歌单** | **用户在“查看推荐歌单/歌曲”中推荐成功后，可将在线歌曲添加到歌单.** |

### 2.2.3 假设和依赖

1.用户必须具备基本的计算机知识.

2. 用户需有一定的使用类似产品的经历

## 2.3 功能需求

### 2.3.1 LWP音乐系统功能

**注册与登录**

- 用户进入系统时，系统提示用户选择注册或登录选项；

- 若用户选择注册选项，系统提示用户填入注册信息；用户输入注册信息：系统验证用户信息，若格式无误，录入信息，执行注册用例，否则提示用户重新输入。

- 若用户选择登录选项，系统提示用户输入登录信息；用户输入登录信息：系统验证用户登录信息，若无误则执行登录用例，否则提示用户重新输入。

**登录成功并使用**

- 若用户选择上传本地音乐，用户可以上传本地歌曲至音乐库内；

- 若用户选择播放本地音乐 ，用户可以播放本地音乐；

- 若用户选择播放歌单中的音乐，用户可以播放账号内歌单中的音乐；

- 若用户选择界面皮肤自定义，用户可以自定义皮肤；

- 若用户选择反馈bug，用户可以填写自己发现的bug并能后续被管理员处理。

**用户选择在线搜索歌曲**

- 若搜索成功，用户可对其进行加入歌单、播放。点赞、收藏、评论操作

**用户选择查看推荐歌曲**

- 若推荐成功，用户可对其进行加入歌单、播放。点赞、收藏、评论操作

## 2.4 非功能需求

### 2.4.1 安全性

| **需求分类** | **定义要素** | **需求内容** |
| --- | --- | --- |
| **系统安全** | **安全机制独立性** | **安全设计和实现应该具有独立性, 不能依赖当前主机的基础安全机制来确保自身和数据不受破环或拒绝服务.** |
| **系统安全** | **安全机制有效性** | **应防止用户绕过其安全控制机制直接尝试访问系统各项功能.** |
| **系统安全** | **访问权限** | **对不同用户的访问权限进行严格的访问控制, 特定权限的用户只能看到和使用特定的界面及相应的功能.** |
| **系统安全** | **输入限制** | **具备输入字符和输入数据的类型, 长度和范围检查功能.** |
| **系统安全** | **权限初始化** | **用户的权限应该符合最小权限原则.** |
| **系统安全** | **并发操作保护** | **允许用户对同一系统资源进行不相冲突的访问操作但应该禁止执行相互矛盾的操作, 如两个用户同时修改一个资源.** |
| **系统安全** | **异常事件** | **应该定义了分级的系统异常事件, 并且根据异常的严重程度分别采用日志记录, 警告提示等方式进行通知. 对部分严重故障自动处理, 使LWP音乐网站系统恢复正常状态或保护现存数据安全.** |
| **系统安全** | **安全日志系统** | **所有的系统出错信息都必须被记录到出错日志中, 信息的格式必须是系统错误号码, 日期, 时间, 错误信息. 和用户相关的每个页面迁移信息都必须被记录到追寻日志中. 信息的格式必须是用户帐号, 日期, 时间, 迁移的页面名. 每次处理前后的消耗的内存和处理时间都必须被记录到性能日志中. 信息的格式必须是日期、时间, 消耗的内存, 处理时间.** |
| **系统安全** | **安全监视系统** | **监视系统必须扫描日志系统如果有任何异常的情况, 必须使用邮件系统向管理员发警告信.** |
| **数据安全** | **数据完整性** | **能够检测到网络设备操作系统, 主机操作系统, 数据库管理系统以及学生信息系统的重要业务数据在传输过程中完整性受到破坏.** |

### 2.4.2 易用性

| **需求分类** | **定义要素** | **需求内容** |
| --- | --- | --- |
| **界面导航要求** | **功能菜单必须按照功能模块, 功能项的分类方法进行组织; 对于用户无权使用的菜单功能应不显示该菜单或将其设置为不可用状态.** | **每个模块相互分隔, 逻辑功能尽可能保持较小的冗余度.** |
| **界面输入要求** | **操作界面必须明确表示出必填的输入信息. 可以按照操作名称建立索引, 点击该事件后可以查看与该事件相关的应急相关信息. 可以按照关键字进行搜索, 系统在界面中将搜索结果分条展示.** | **每个可能的输入都有确定的返回值, 如果出现异常会及时报错并提供正确的帮助信息.** |
| **界面提示要求** | **发现用户提交有误信息, 必须以弹出窗口的形式明确提示用户错误的原因, 并把界面控制焦点置于发生错误的控件对象上; 在导致数据发生变化的操作执行之前, 应该弹出提示窗口供用户确认; 当学生信息系统正在执行用户提交的请求而无法返回时, 必须明确表示平台处于繁忙阶段, 并可以让用户取消操作.** | **LWP音乐系统的页面上必须有相关的帮助信息, 用户可以使用它们来获得必要的信息.** |

### 2.4.3 可靠性

* 可用性

离线也可用本系统播放本地音乐

* 系统发生严重错误的平均时间间隔

系统发生严重错误的平均时间间隔应该大于4000小时.

### 2.4.4 性能

* 最大的并发人数

LWP音乐系统的最大并发访问数应该为100. 在这个范围内, 系统应该能够很好的工作.

* 最大的事务处理时间

LWP音乐系统的用户事务的最大处理时间是5秒, 如果超过这个时间系统应该自动结束用户的事务处理.

## 2.5. 运行环境规定

* Web 服务器软件

系统必须使用支持 J2EE 规范的 Web 服务器软件.

* 平台要求

系统必须在操作系统为微软 windows xp以上或 Linux(Unix)的数据库服务器上顺利运行.

* Web 浏览器

系统的用户界面必须在IE 8.0以上或 Chrome70.0 以上的浏览器正常显示.

## 2.6. 验收验证标准

### 2.6.1 界面验收标准

| **序号** | **界面名称** | **界面描述及预期效果** |
| --- | --- | --- |
| **1** | **系统界面** | **显示顶栏、系统logo、前进后退按钮、音乐搜索框、用户头像按钮、侧边栏、音乐发现按钮、视频按钮、收藏歌曲单按钮、每日推荐歌曲按钮、音乐发现页面下的内容、底部播放栏、播放按钮、切歌按钮、音乐大小按钮、播放列表按钮.** |
| **2** | **音乐发现界面** | **显示个性推荐页面按钮以及个性推荐界面下推荐的歌曲、推荐的歌单封面与名字，显是歌单界面按钮、排行榜界面按钮、歌手界面按钮.** |
| **3** | **歌单推荐界面** | **显示歌单推荐的分类按钮、推荐的歌单的封面与名字，点击不同推荐分类的歌单转换成对应的歌单界面.** |
| **4** | **歌单界面** | **显示歌单的封面、歌单作者、播放歌曲按钮、收藏歌单按钮、分享按钮、歌单内的歌曲列表、歌单的评论、歌单的收藏着.** |
| **5** | **排行榜界面** | **根据收藏数显示官方的歌曲排行榜、包含歌曲排行信息、可以拖动页面查看不同的排行榜、可以点击进排行榜的歌单界面.** |
| **6** | **歌手界面** | **显示不同歌手的语种的按钮、根据男女歌手进行分类、还可以根据歌手的名字首字母进行分类、界面内显示歌手名字以及照片、可以点击歌手的名字或者照片查看本系统收录的该歌手的所有歌曲并进行播放.** |
| **7** | **视频与mv界面** | **显示lwp音乐系统收录的视频与mv，点击视频或mv按钮，可以进入视频界面或mv界面、点击对应的视频或者mv封面或名字后能够进入视频播放页界面.** |
| **8** | **视频与mv播放界面** | **可以播放视频与mv、能够在该视频与mv下进行评论，能够查看视频与mv的发布信息，可以对视频进行点赞收藏与分享、能够查看别人的评论、可以推荐相关的视频或者mv.** |
| **9** | **歌曲收藏界面** | **按照专辑、歌手、视频的分类显示用户收藏的歌曲，要包含专辑的名字与封面、歌手的名字信息，视频的名字与封面.** |
| **10** | **每日推荐歌曲界面** | **显示系统根据用户习惯每天推荐的歌曲名字、歌手、专辑、歌曲时长、推荐更新时间.** |
| **11** | **用户界面** | **显示用户头像、用户名称、退出登录按钮、用户动态数、用户关注数、粉丝数、用户创建的歌单封面与名字、用户收藏的歌单封面与名字.** |

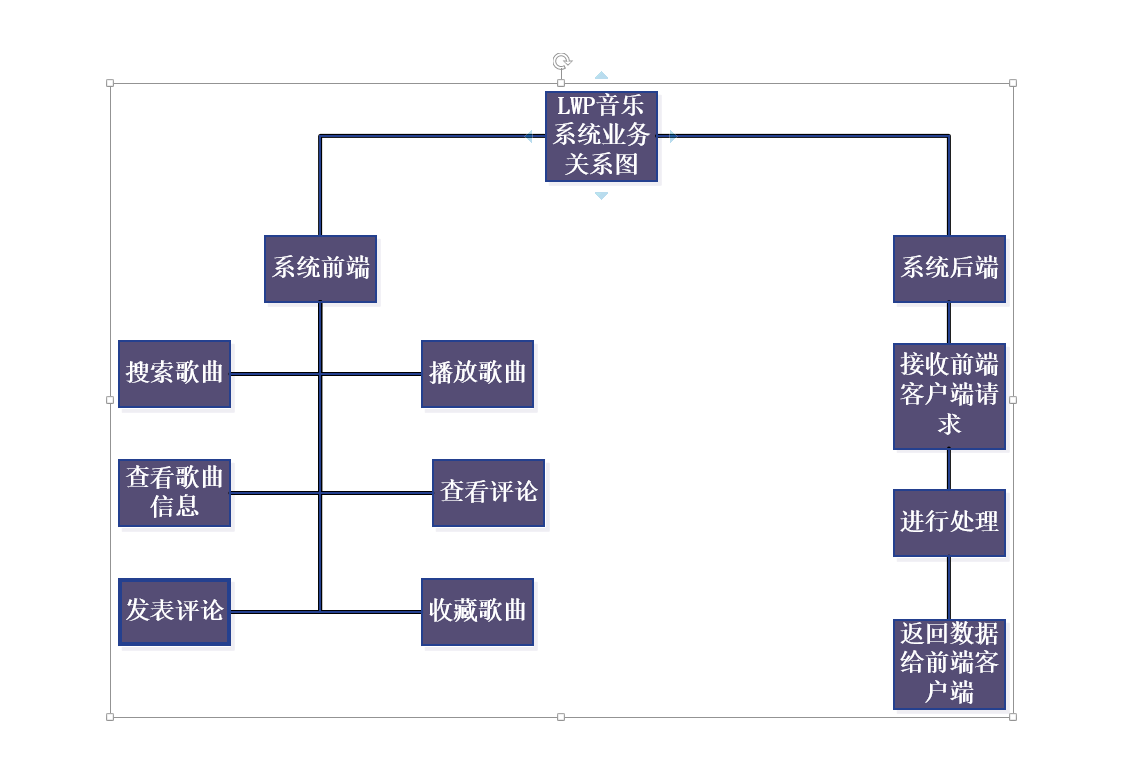
### 2.6.2 功能验收标准

| **序号** | **功能名称** | **详细操作** | **预期效果** |
| --- | --- | --- | --- |
| **1** | **登录与注册** | **可点击右上角按钮进行登录** | **显示登录与注册框，提示用户输入登陆或注册信息并进行验证，登陆后右上角显示用户头像与名字.** |
| **2** | **音乐播放** | **可点击歌单或者搜索到的音乐名字进行选择** | **点击后下方播放栏跳转到对应歌曲播放信息并进行播放.** |
| **3** | **音乐切换** | **点击上一首下一首切换按钮实现音乐切换.** | **点击按钮后切换上一首或下一首播放的音乐.** |
| **4** | **收藏音乐** | **点击喜欢按钮收藏音乐** | **点击后按钮颜色变化, 并加入用户默认的喜欢收藏歌单.** |
| **5** | **取消收藏** | **点击喜欢按钮取消收藏音乐** | **点击后按钮颜色变化, 并将歌曲从用户默认的喜欢收藏歌单中删除.** |
| **6** | **推荐歌曲** | **点击每日推荐歌曲按钮查看推荐歌曲列表.** | **点击后按钮后跳转到每日歌曲推荐列表，显示系统推荐的歌曲列表.** |
| **7** | **评论歌曲** | **点击左下角歌曲封面按钮进入歌曲歌词与评论页，点击播放栏上方的评论发表框，进行评论** | **跳转至歌曲歌词与评论页，点击评论框按钮后跳出评论框，输入完评论后点击评论按钮即可.** |
| **8** | **歌词查看** | **点击左下角歌曲封面按钮进入歌曲歌词与评论页** | **进入歌曲歌词与评论页，即可查看歌词.** |
| **9** | **评论查看** | **点击左下角歌曲封面按钮进入歌曲歌词与评论页** | **进入歌曲歌词与评论页，即可查看该歌曲下的热门评论与所有评论.** |
| **10** | **视频与mv播放** | **点击视频或mv封面，进入视频或mv的播放页，点击播放** | **点击封面后，跳转到对应的播放页，点击播放.** |
| **11** | **视频与mv评论查看** | **点击视频或mv封面，进入视频或mv的播放页，下拉滚动条** | **点击封面后，跳转到对应的播放页，在页面下方显示评论.** |
| **12** | **用户信息查看** | **点击右上角用户头像按钮** | **点击用户头像按钮后，进入用户信息页，显示有用户创建与收藏的歌单信息以及用户的动态、关注、粉丝数.** |

# 三.系统设计说明书

## 3.1 总体设计

### 3.1.1 系统业务关系图



### 3．1.2 系统后端请求处理流程图

## 3.2 系统模块设计

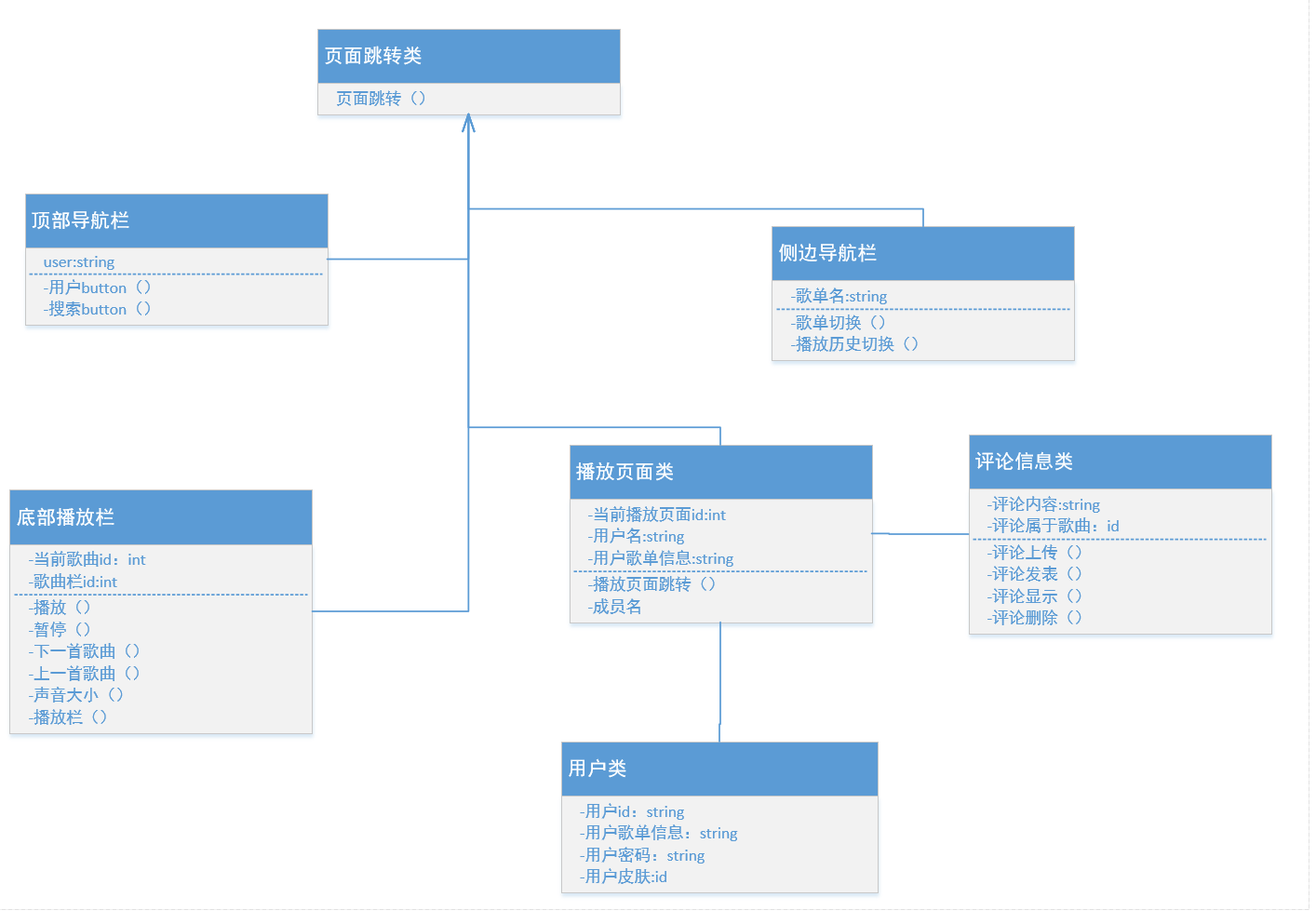
由需求分析与系统业务关系图产生，对目前系统总结出3个功能模块：

1.ui模块：系统界面的各个按钮部分，负责对用户的点击进行各种信息反馈。

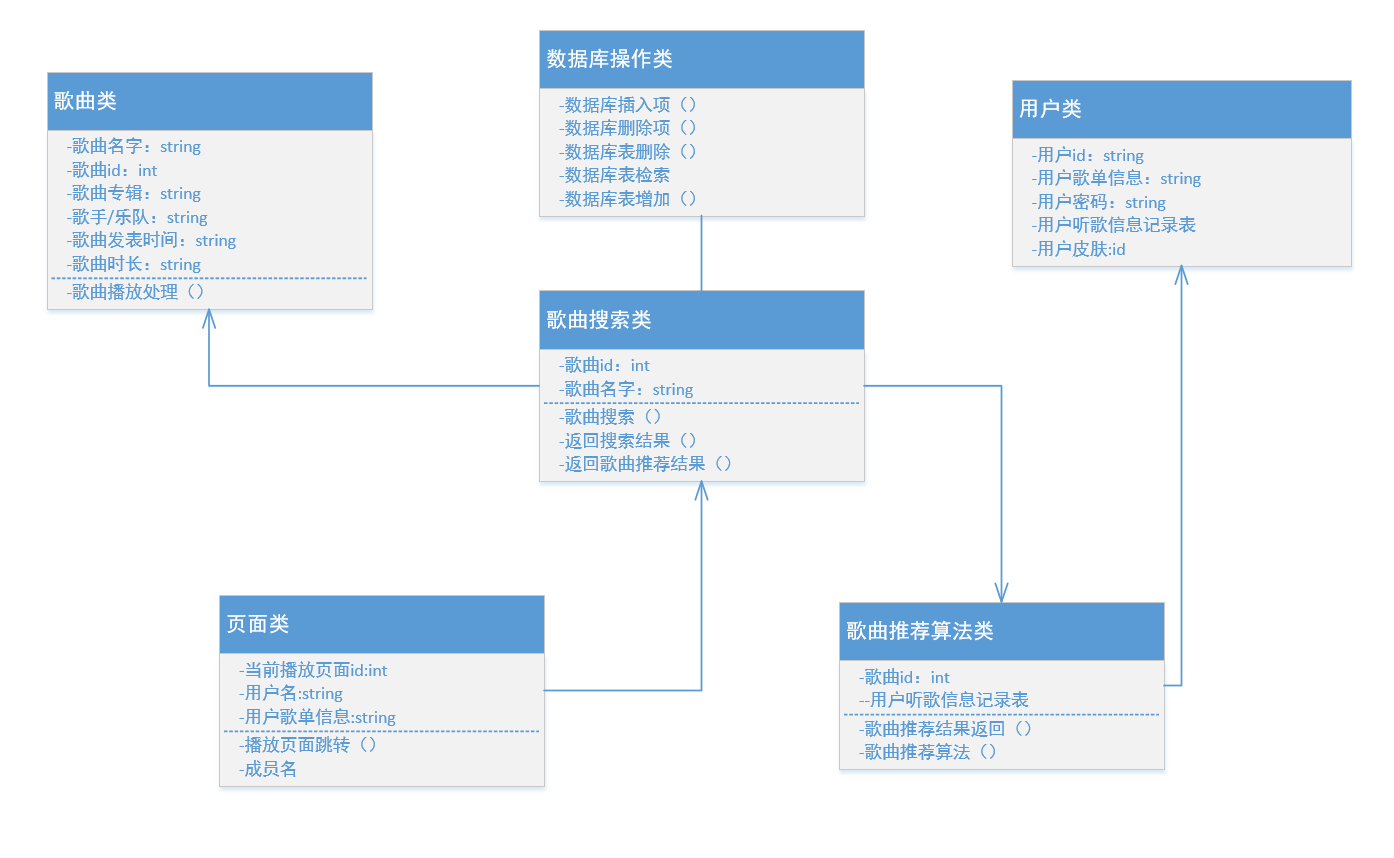
2.搜索模块：用于向数据库请求搜索歌曲、评论、用户信息等操作。

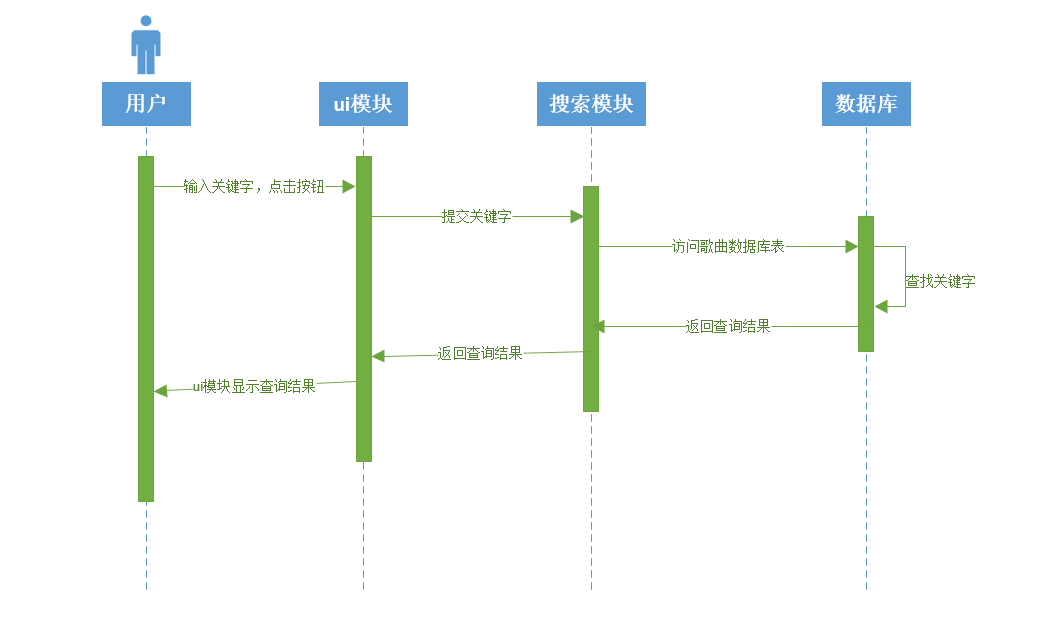
## 3.3 功能模块设计

### 3.3.1 ui模块

**ui模块类图**

### 3.3.2 搜索模块

**类图**

**时序图**

### 四.项目实现

## 4.1 数据准备

### 4.1.1 数据来源说明

一开始开启本项目，对于歌曲数据来源的想法是小组团队上网搜索下载歌曲上传到服务器上，但随着项目的开发逐步落实，确定了本想法的可用性极差。

随后小组成员便开始思考，本系统的开发初衷是实现一个音乐播放系统，对于歌曲数据的收集并不是本系统的侧重点，故我们推翻了一开始的想法，将系统的后端实现与设计采用调用网络上开源的音乐api服务的形式，获取音乐数据，对得到的数据经过后端服务处理成前端需要的数据，再返回给前端。故本系统不需要使用到数据库。

但本系统需要注意的点是，对各开源音乐api的参数进行整理与记录。本系统采用的是网易云api的接口服务。

### 4.1.2 系统后端接口调用参数说明记录

说明：该部分为本系统所使用到的网易云api的调用参数说明，记录该参数说明是为了实现后端时更方便的进行查阅。

1.注册(修改密码)

说明 : 调用此接口 ,传入手机号码和验证码,密码,昵称。

必选参数 :

captcha: 验证码

phone : 手机号码

password: 密码

nickname: 昵称

2.初始化昵称

说明 : 刚注册的账号(需登录),调用此接口 ,可初始化昵称。

必选参数 : nickname : 昵称

3.获取用户信息 , 歌单，收藏，mv, dj 数量

说明 : 登录后调用此接口 , 可以获取用户信息。

4.更新歌单

说明 : 登录后调用此接口,可以更新用户歌单。

必选参数 :

id:歌单id

name:歌单名字

desc:歌单描述

tags:歌单tag ,多个用 `;` 隔开,只能用官方规定标签

5.更新歌单描述

说明 : 登录后调用此接口,可以单独更新用户歌单描述

必选参数 :

id:歌单id

desc:歌单描述

6.对歌单添加或删除歌曲

说明 : 调用此接口 , 可以添加歌曲到歌单或者从歌单删除某首歌曲 ( 需要登录 )

必选参数 :

op: 从歌单增加单曲为 add, 删除为 del

pid: 歌单 id tracks: 歌曲 id,可多个,用逗号隔开

7.新歌速递

说明 : 调用此接口 , 可获取新歌速递

必选参数 :

type: 地区类型 id,对应以下:

全部:0

华语:7

欧美:96

日本:8

韩国:16

8.喜欢音乐

说明 : 调用此接口 , 传入音乐 id, 可喜欢该音乐

必选参数 : id: 歌曲 id

可选参数 : like: 布尔值 , 默认为 true 即喜欢 , 若传 false, 则取消喜欢

9.推荐歌单

说明 : 调用此接口 , 可获取推荐歌单

可选参数 : limit: 取出数量 , 默认为 30 (不支持 offset)

10.歌手榜

说明 : 调用此接口 , 可获取排行榜中的歌手榜

可选参数 :

type : 地区

1: 华语

2: 欧美

3: 韩国

4: 日本

11.获取音乐 url

说明 : 使用歌单详情接口后 , 能得到的音乐的 id, 但不能得到的音乐 url, 调用此接口, 传入的音乐 id( 可多个 , 用逗号隔开 ), 可以获取对应的音乐的 url,未登录状态或者非会员返回试听片段(返回字段包含被截取的正常歌曲的开始时间和结束时间)

注 : 部分用户反馈获取的 url 会 403,hwaphon找到的解决方案是当获取到音乐的 id 后，将 https://music.163.com/song/media/outer/url?id=id.mp3 以 src 赋予 Audio 即可播放

必选参数 : id : 音乐 id

可选参数 : br: 码率,默认设置了 999000 即最大码率,如果要 320k 则可设置为 320000,其他类推

12.搜索

说明 : 调用此接口 , 传入搜索关键词可以搜索该音乐 / 专辑 / 歌手 / 歌单 / 用户 , 关键词可以多个 , 以空格隔开 , 如 " 周杰伦 搁浅 "( 不需要登录 ), 搜索获取的 mp3url 不能直接用 , 可通过 /song/url 接口传入歌曲 id 获取具体的播放链接

必选参数 : keywords : 关键词

可选参数 : limit : 返回数量 , 默认为 30 offset : 偏移数量，用于分页 , 如 : 如 :( 页数 -1)\*30, 其中 30 为 limit 的值 , 默认为 0

type: 搜索类型；默认为 1 即单曲 , 取值意义 : 1: 单曲, 10: 专辑, 100: 歌手, 1000: 歌单, 1002: 用户, 1004: MV, 1006: 歌词, 1009: 电台, 1014: 视频, 1018:综合

13.歌单收藏者

说明 : 调用此接口 , 传入歌单 id 可获取歌单的所有收藏者

必选参数 :

id : 歌单 id

可选参数 : limit: 取出评论数量 , 默认为 20

offset: 偏移数量 , 用于分页 , 如 :( 评论页数 -1)\*20, 其中 20 为 limit 的值

14.获取歌词

说明 : 调用此接口 , 传入音乐 id 可获得对应音乐的歌词 ( 不需要登录 )

必选参数 : id: 音乐 id

15.收藏/取消收藏歌单

说明 : 调用此接口 , 传入类型和歌单 id 可收藏歌单或者取消收藏歌单

必选参数 :

t : 类型,1:收藏,2:取消收藏 id : 歌单 id

16.歌曲评论

说明 : 调用此接口 , 传入音乐 id 和 limit 参数 , 可获得该音乐的所有评论 ( 不需要登录 )

必选参数 : id: 音乐 id

可选参数 : limit: 取出评论数量 , 默认为 20

offset: 偏移数量 , 用于分页 , 如 :( 评论页数 -1)\*20, 其中 20 为 limit 的值

before: 分页参数,取上一页最后一项的 time 获取下一页数据(获取超过5000条评论的时候需要用到)

17.歌单评论

说明 : 调用此接口 , 传入音乐 id 和 limit 参数 , 可获得该歌单的所有评论 ( 不需要 登录 )

必选参数 : id: 歌单 id

可选参数 : limit: 取出评论数量 , 默认为 20

offset: 偏移数量 , 用于分页 , 如 :( 评论页数 -1)\*20, 其中 20 为 limit 的值

before: 分页参数,取上一页最后一项的 time 获取下一页数据(获取超过5000条评论的时候需要用到)

18.歌手分类列表

说明 : 调用此接口 , 可获取歌手分类列表

可选参数 : limit: 返回数量 , 默认为 30

offset: 偏移数量 , 用于分页 , 如 :( 评论页数 -1)\*20, 其中 20 为 limit 的值

initial: 按首字母索引查找参数

type：取值：

-1：全部

-2：男歌手

-3：女歌手

-4：乐队

Area：取值：

-1：全部

7华语

96欧美

8:日本

16韩国

0:其他

19.mv评论

说明 : 调用此接口 , 传入音乐 id 和 limit 参数 , 可获得该mv的所有评论 ( 不需要 登录 )

必选参数 : id: mv id

可选参数 : limit: 取出评论数量 , 默认为 20

offset: 偏移数量 , 用于分页 , 如 :( 评论页数 -1)\*20, 其中 20 为 limit 的值

before: 分页参数,取上一页最后一项的 time 获取下一页数据(获取超过5000条评论的时候需要用到)

20.全部mv

说明 : 调用此接口 , 可获取全部mv

可选参数 : limit: 取出量 , 默认为 30

offset: 偏移数量 , 用于分页 , 如 :( 评论页数 -1)\*20, 其中 20 为 limit 的值

order：排序，可选值为上升最快，最热，最新，不填则为上升最快

21.获取mv数据

说明 : 调用此接口 , 传入mvid，可获取对应mv数据，包含mv没名字，歌手，发布时间，mv视频地址等。

必选参数 : id: mvid

返回数据如图：

22.mv地址

说明 : 调用此接口 , 传入mvid，可获取对应mv播放地址

必选参数：id：mvid

可选参数：r：分辨率，默认1080

## 4.2 前端实现

### 4.2.1 Vue.js介绍

本项目前端采用vue框架+css+html语言实现，Vue (读音 /vjuː/，类似于 view) 是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与现代化工具链以及各种支持类库结合使用时，Vue 也完全能够为复杂的单页应用提供驱动。

Vue.js 使用了基于 HTML 的模板语法，允许开发者声明式地将 DOM 绑定至底层组件实例的数据。所有 Vue.js 的模板都是合法的 HTML，所以能被遵循规范的浏览器和 HTML 解析器解析。

在底层的实现上，Vue 将模板编译成虚拟 DOM 渲染函数。结合响应性系统，Vue 能够智能地计算出最少需要重新渲染多少组件，并把 DOM 操作次数减到最少。

总结来说，使用vue框架来开发本项目前端将会十分轻便。

### 4.2.2 前端项目结构

本前端项目结构如图4.1 项目结构总体图所示。

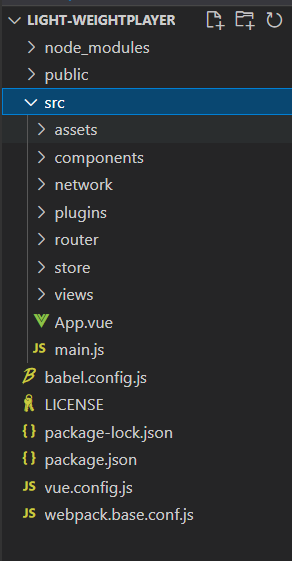
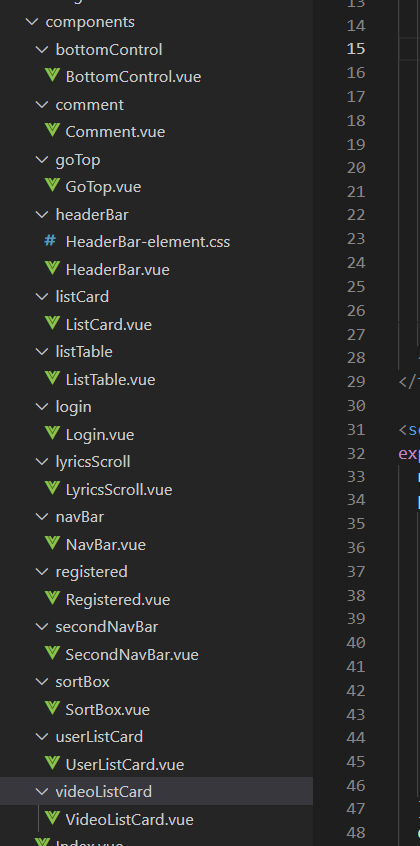
其中，assets是本前端项目中用到的图片资源以及css描述等文件。Components文件夹中定义了本项目所用到的各种可重用的自定义组件，如图4.2 components组件结构图所示。

图4.1 项目结构总体图 图4.2 components组件结构图

Views文件夹中存放的是对具体页面的封装与页面之间的跳转逻辑关系，详细的目录结构如图4.3 views组件结构图所示。

### 4.2.3 router设置

router，即“路由”。路由是一个网络工程里面的术语。James F.Kurose,Keith W.Ross所写的《计算机网络》中有讲到：路由（routing）是指分组从源到目的地时，决定端到端路径的网络范围的进程

在WEB开发的世界里面，路由其实指的就是控制我们在浏览器中输入的URL应该走入哪个页面中的一个组件。

vue-router 默认 hash 模式 —— 使用 URL 的 hash 来模拟一个完整的 URL，于是当 URL 改变时，页面不会重新加载。但hash模式有一个很大的问题，使用hash模式的话，页面地址会加上一个＃号，变得不雅观，这里我们用到路由的history模式。这种模式充分利用 history.pushState API 来完成 URL 跳转而无须重新加载页面。

Router文件夹的部分代码配置如下：

Vue.use(VueRouter)

const routes = [

    { path: '/', redirect: '/index' },

    {

        path: '/index',

        component: Index,

        redirect: '/discover',

        children: [

            {

                path: '/discover',

                component: Discover,

                redirect: '/discover/recommend',

                children: [

                    { path: '/discover/recommend', component: Recommend },

                    {

                        path: '/discover/musiclist',

                        component: MusicList,

                        redirect: '/discover/musiclist/musiclistindex',

                        children: [

                            { path: '/discover/musiclist/musiclistindex', component: MusicListIndex },

                            { path: '/discover/musiclist/musiclisthighquality', component: MusicListHighquality },

                        ]

                    },

                    { path: '/discover/ranking', component: Ranking },

                    { path: '/discover/singer', component: Singer },

                ]

            },

            {

                path: '/video', component: Video,

                redirect: '/video/videolist',

                children: [

                    { path: '/video/videolist', component: VideoList },

                    { path: '/video/mvlist', component: MvList },

                ]

            },

            { path: '/musiclistdetail/:id', name: 'musicListDetail', component: MusicListDetail },

            { path: '/singerdetail/:id', name: 'singerDetail', component: SingerDetail },

            { path: '/album/:id', name: 'album', component: Album },

            { path: '/videodetail/:id/:type', name: 'videoDetail',

    }

]

其中，path定义的是所访问的组件页面的页面路径。Component组件则是在文件开头引入。

## 4.3 后端实现

### 4.3.1 逻辑实现

本后端项目的基本逻辑如下：

1. 接受客户端请求
2. 预处理:放行请求，允许跨域,拿出cookie（app.js)
3. 构造伪请求，封装必要数据(module,util/request.js)
4. 将数据进行加密，构造特定的请求参数(util/crypto.js)
5. 向网易云发送请求(util/request.js)
6. 解析返回数据，将数据返回给客户端，对于登录请求，还要写入cookie

### 4.3.2 后端构建

本后端采用gin来处理路由，以singo为脚架快速搭建web应用程序，采用asmcos/request发送请求。

下面为项目构建的部分重点代码说明：

1. 请求数据封装

以手机登录接口为例，将客户端发送的请求绑定到结构体中。

type LoginCellphoneService struct {

    Phone       string `json:"phone" form:"phone"`

    Countrycode string `json:"countrycode" form:"countrycode"`

    Password    string `json:"password" form:"password"`

    Md5password string `json:"md5\_password" form:"md5\_password"`

}

接着是功能函数：

func (service \*LoginCellphoneService) LoginCellphone(c \*gin.Context) map[string]interface{} {

    // 获得所有cookie

    cookies := c.Request.Cookies()

    cookiesOS := &http.Cookie{Name: "os", Value: "pc"}

    cookies = append(cookies, cookiesOS)

    options := &util.Options{

        Crypto:  "weapi",

        Ua:      "pc",

        Cookies: cookies,

    }

    data := make(map[string]string)

    data["phone"] = service.Phone

    if service.Countrycode != "" {

        data["countrycode"] = service.Countrycode

    }

    if service.Password != "" {

        h := md5.New()

        h.Write([]byte(service.Password))

        data["password"] = hex.EncodeToString(h.Sum(nil))

    } else {

        data["password"] = service.Md5password

    }

    data["rememberLogin"] = "true"

    reBody, cookies := util.CreateRequest("POST", `https://music.163.com/weapi/login/cellphone`, data, options)

    cookiesStr := ""

    for \_, cookie := range cookies {

        if cookiesStr != "" {

            cookiesStr = cookiesStr + ";"

        }

        cookiesStr = cookiesStr + cookie.String()

        c.SetCookie(cookie.Name, cookie.Value, 60\*60\*24, "", cookie.Domain, false, false)

    }

    reBody["cookie"] = cookiesStr

    return reBody

}

该功能函数首先获得客户端的所有cookie，然后构造请求参数，util.Options为请求选项的封装，data为请求的body的所需原数据，随后对密码进行md5加密，然后将数据发往request 包括 请求方法，连接，数据，请求选项 返回网易云的数据返回和set-cookie，最后将cookie写入。

2.请求函数构建

接着是请求函数的实现，首先我们定义的请求选项的结构体，代码如下：

type Options struct {

    Crypto  string

    Ua      string

    Cookies []\*http.Cookie

    Token   string

    Url     string

}

随后，初始化一个请求对象，设置请求头，定义返回对象，将cookie写入请求体中并获取部分cookie的值。

req := requests.Requests()

    req.Header.Set("User-Agent", chooseUserAgent(options.Ua))

    csrfToken := ""

    music\_U := ""

    answer := map[string]interface{}{}

    if method == "POST" {

        req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

    }

    if strings.Contains(url, "music.163.com") {

        req.Header.Set("Referer", "https://music.163.com")

    }

    if options.Cookies != nil {

        for \_, cookie := range options.Cookies {

            req.SetCookie(cookie)

            if cookie.Name == "\_\_csrf" {

                csrfToken = cookie.Value

            }

            if cookie.Name == "MUSIC\_U" {

                music\_U = cookie.Value

            }

        }

    }

紧接着根据不同的请求类型进入不同的加密函数，将data数据写入eapiData。

if options.Crypto == "weapi" {

        data["csrf\_token"] = csrfToken

        data = Weapi(data)

        reg, \_ := regexp.Compile(`/\w\*api/`)

        url = reg.ReplaceAllString(url, "/weapi/")

    } else if options.Crypto == "linuxapi" {

        linuxApiData := make(map[string]interface{}, 3)

        linuxApiData["method"] = method

        reg, \_ := regexp.Compile(`/\w\*api/`)

        linuxApiData["url"] = reg.ReplaceAllString(url, "/api/")

        linuxApiData["params"] = data

        data = Linuxapi(linuxApiData)

        req.Header.Set("User-Agent", "Mozilla/5.0 (X11; Linux x86\_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.90 Safari/537.36")

        url = "https://music.163.com/api/linux/forward"

    } else if options.Crypto == "eapi" {

        eapiData := make(map[string]interface{})

        for key, value := range data {

            eapiData[key] = value

        }

然后将header里的数据写入cookie，将header写入eapiData

for key, value := range header {

            req.SetCookie(&http.Cookie{Name: key, Value: value, Path: "/"})

        }

        eapiData["header"] = header

        data = Eapi(options.Url, eapiData)

        reg, \_ := regexp.Compile(`/\w\*api/`)

        url = reg.ReplaceAllString(url, "/eapi/")

    }

若发生请求错误，将错误写入相应的错误码。

if err != nil {

        answer["code"] = 520

        answer["err"] = err.Error()

        return answer, nil

    }

接着获取返回的cookie以及body，对数据尝试zlib解压。吧json字符串转化为对象写入如answer，最后查询answeryouwucode字段，无则写入200，避免返回值中无code字段。

cookies := resp.Cookies()

    body := resp.Content()

    b := bytes.NewReader(body)

    var out bytes.Buffer

    r, err := zlib.NewReader(b)

    // 数据被压缩 进行解码

    if err == nil {

        io.Copy(&out, r)

        body = out.Bytes()

    }

    err = json.Unmarshal(body, &answer)

     if err != nil {

        if strings.Index(string(body), "<!DOCTYPE html>") != -1 {

            answer["code"] = 200

            answer["html"] = string(body)

            return answer, cookies

        }

        answer["code"] = 500

        answer["err"] = err.Error()

        fmt.Println(string(body))

        return answer, nil

    }

    if \_, ok := answer["code"]; !ok {

        answer["code"] = 200

    }

    return answer, cookies

# 五.测试报告

## 5.1 引言

### 5.1.1 编写目的

本测试报告为LWP音乐系统项目的测试报告, 目的在于总结测试阶段的测试以及分析测试结果, 描述系统是否符合需求. 预期参考人员包括用户、测试人员、开发人员、项目管理者、其他质量管理人员和需要阅读本报告的高层经理.

### 5.1.2 项目背景

1.项目名称：LWP音乐系统

2.项目开发者：陈柏钧、郑宗汕、陈惟逸、冯珞轩、陈昊阳、张钦博、文柯力

3.项目用户：喜好听歌的用户

### 5.1.3 系统简介

在人类社会的各个领域中，计算机科学技术发挥着越来越重要的作用。个人用户对于听音乐的需求也越来越大，用户对音乐检索与音乐播放需求越来越大。

​故我们力求与开发一个LWP音乐系统，能够使得用户需要时检索并播放需求的音乐，随时分享自己的听歌感受，创建属于自己的歌单，网能够根据用户的听歌喜好分析并推荐音乐。为用户创造一个良好的听歌平台。

## 5.2 测试任务概述

### 5.2.1 目标

本项目的主要需求为:

1. 登录与注册使用：

2.用户选择在线搜索歌曲

3.用户选择查看推荐歌曲

4.用户可以播放所选择的歌曲，并查看当前歌曲的信息、歌词、评论留言等信息

5.用户播放歌曲时可以控制歌曲播放的进度，声音大小

### 5.2.2 运行环境

* 平台要求

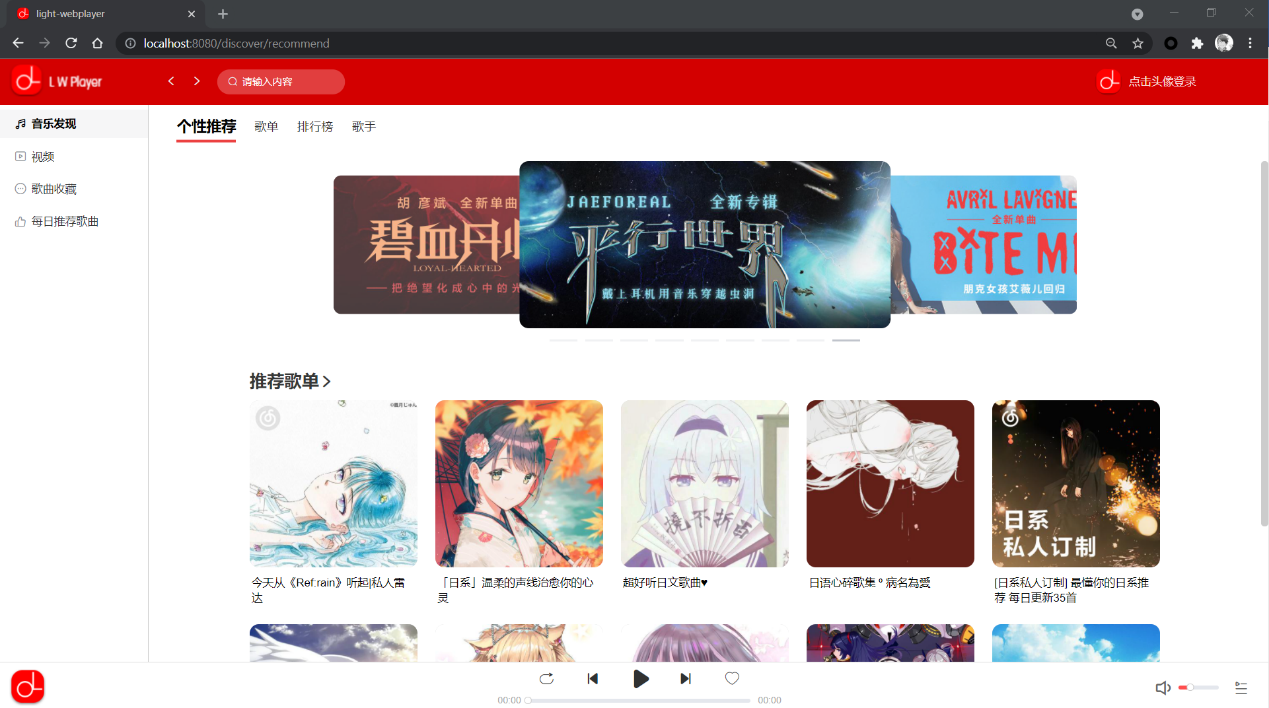
系统必须在操作系统为微软 windows xp以上或 Linux(Unix)的数据库服务器上顺利运行.

* Web 浏览器

系统的用户界面必须在IE 8.0以上或 Chrome70.0 以上的浏览器正常显示.

## 5.3 测试演示

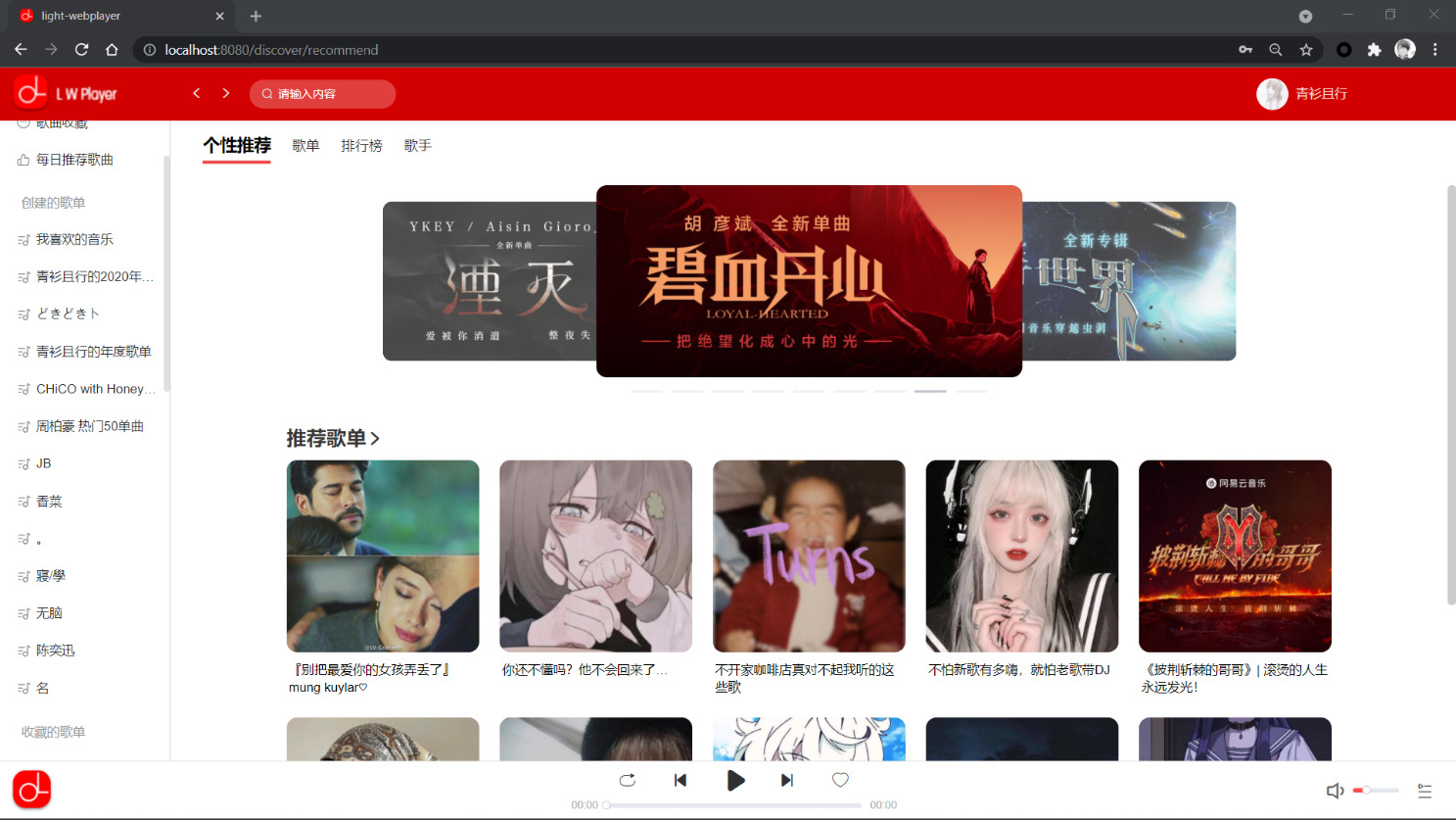
### 5.3.1 lwp音乐系统主页效果测试

未登录时，lwp音乐系统的登陆界面如下：

此时，只有音乐发现界面下的歌曲才可以播放且点击，点击视频、歌曲收藏、每日推荐歌曲按钮时钧会提醒用户登录后才可执行。如下图所示：

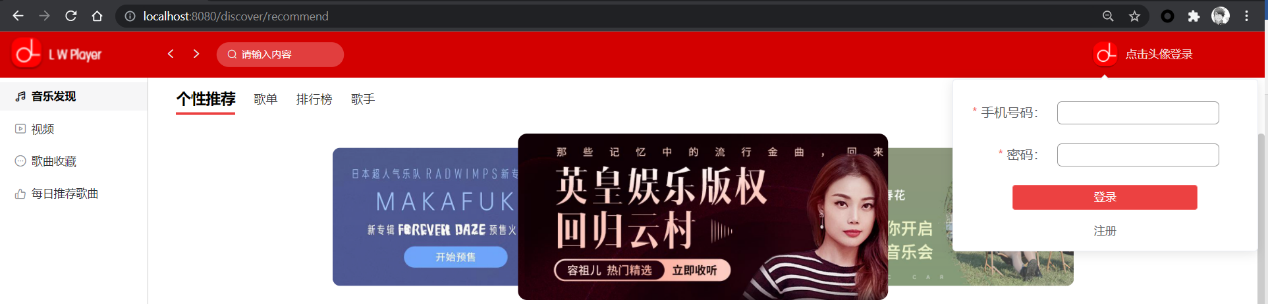
该图是连续点击左侧栏的视频、歌曲收藏、每日推荐按钮后连续弹出的的三个错误提醒。

若单独点击时，只会弹出一个错误提醒，如下图：

 登陆后，系统的界面如下：

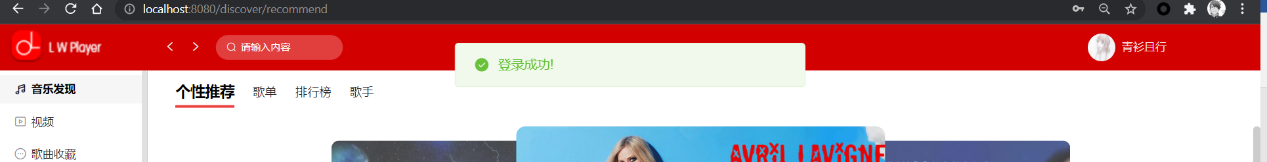
可以看到，系统的左侧多出了创建的歌单、收藏的歌单两栏抽屉式菜单。实现了系统登陆后读取用户歌单的功能。

### 5.3.2登录窗口界面效果测试

点击右上角的头像登录logo后，可以看到登录窗口弹出。

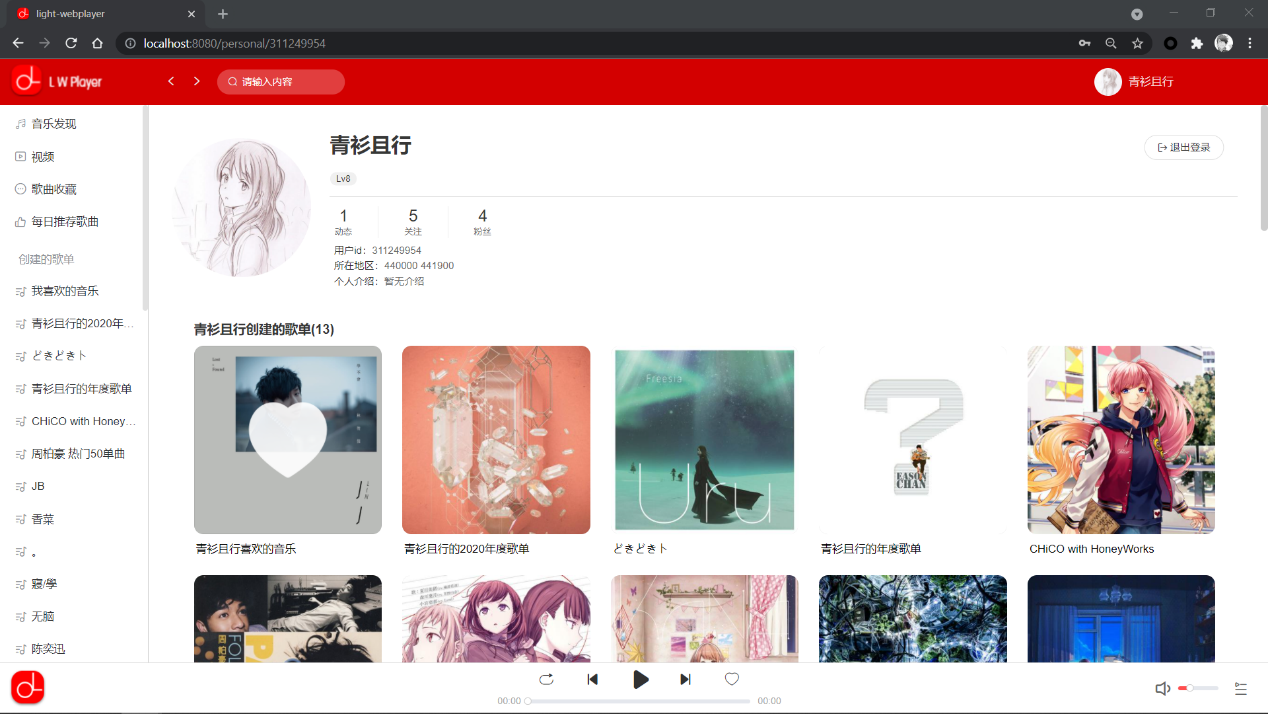
接下来测试登录验证功能，输入错误的密码时，系统页面显示如下：

系统弹出密码错误弹窗。

输入正确密码登陆后，系统显示如下：

系统弹出登录成功提示弹窗。以上，系统登录功能测试完毕

### 5.3.3 用户界面效果测试

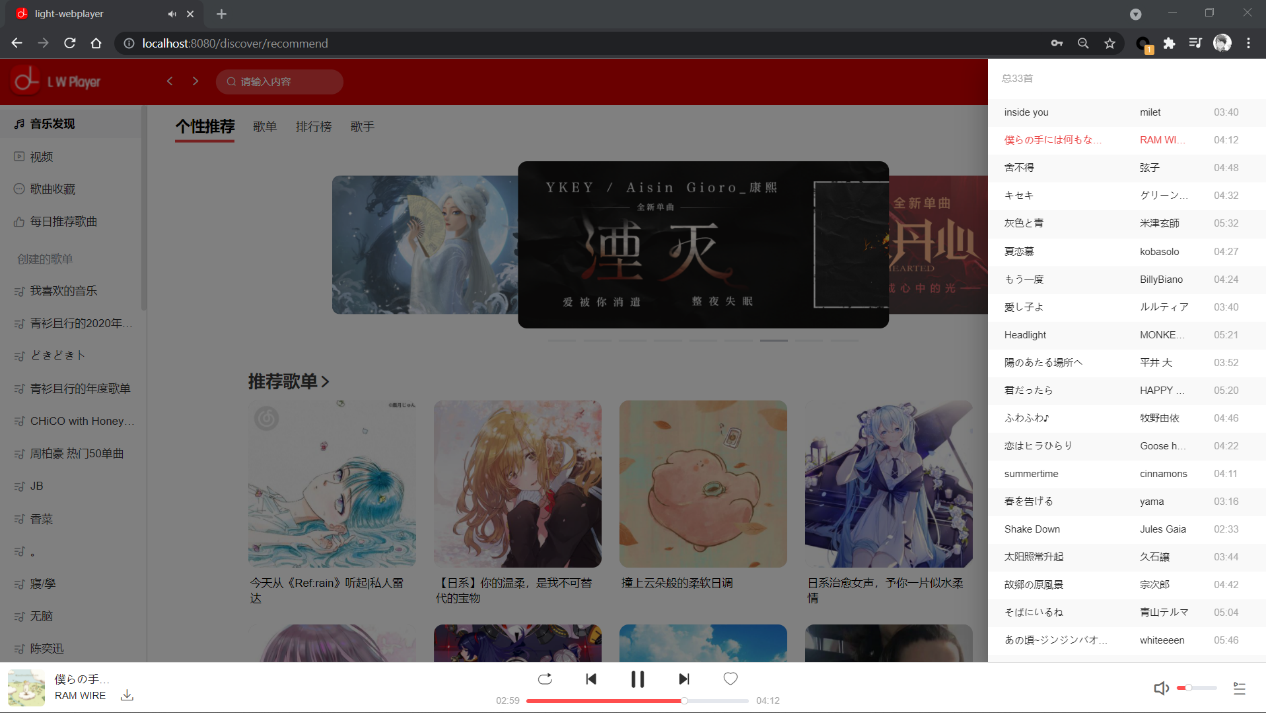
点击右上角的用户头像，测试是否能够看到用户的信息。效果图如下：

可以看到，用户的id、动态、关注数、粉丝数、歌单等信息均已显示。

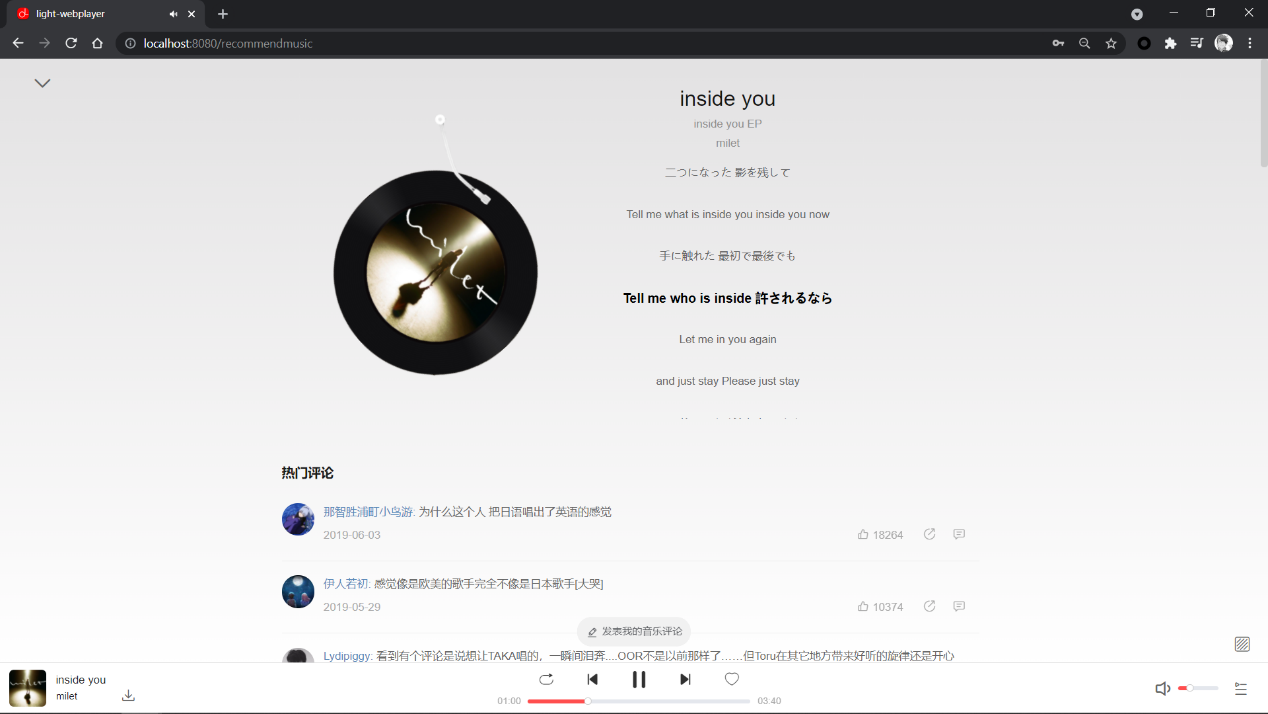
### 5.3.4歌曲播放测试

进入每日推荐歌曲页，选一首歌曲点击播放，播放页面如下：

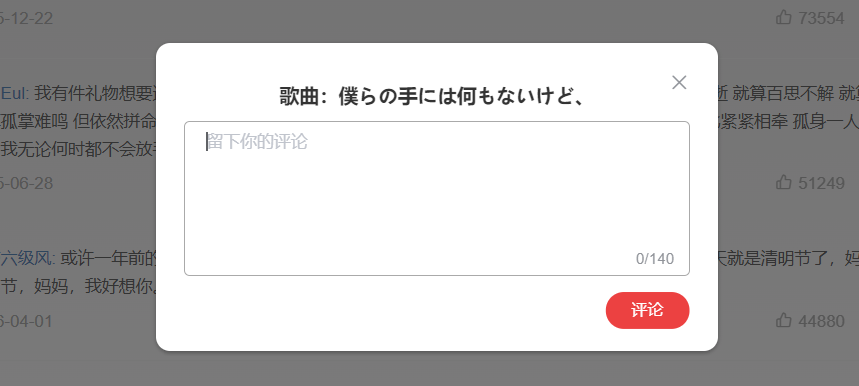
可以看到歌曲已经播放，经过测试，歌曲正常播放，拖动下侧播放栏与音乐大小条后，改变歌曲播放进度和调整音量大小的功能均成功实现。同时，上一首与下一首歌的切换功能钮也能正常实现。

点击右下角的播放栏按钮，可以看到当前播放的歌曲历史，以及即将要播放的歌曲，如图所示：

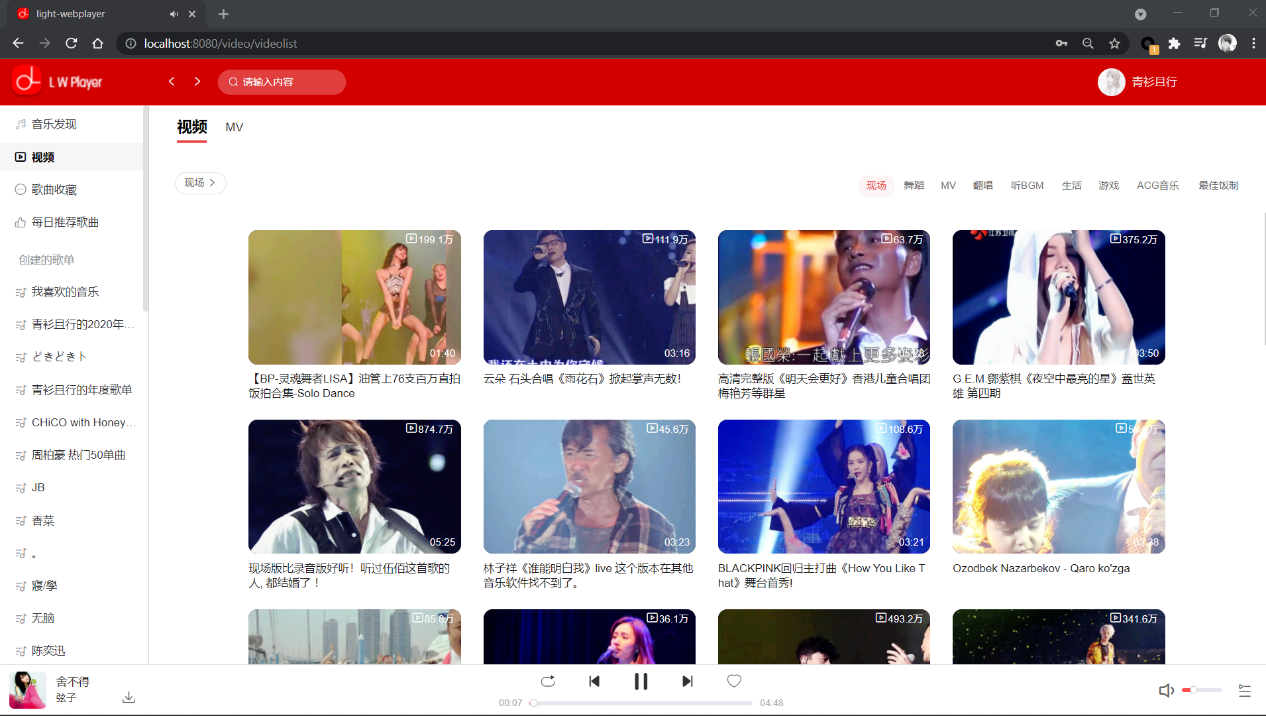
### 5.3.5 歌词与评论界面效果测试

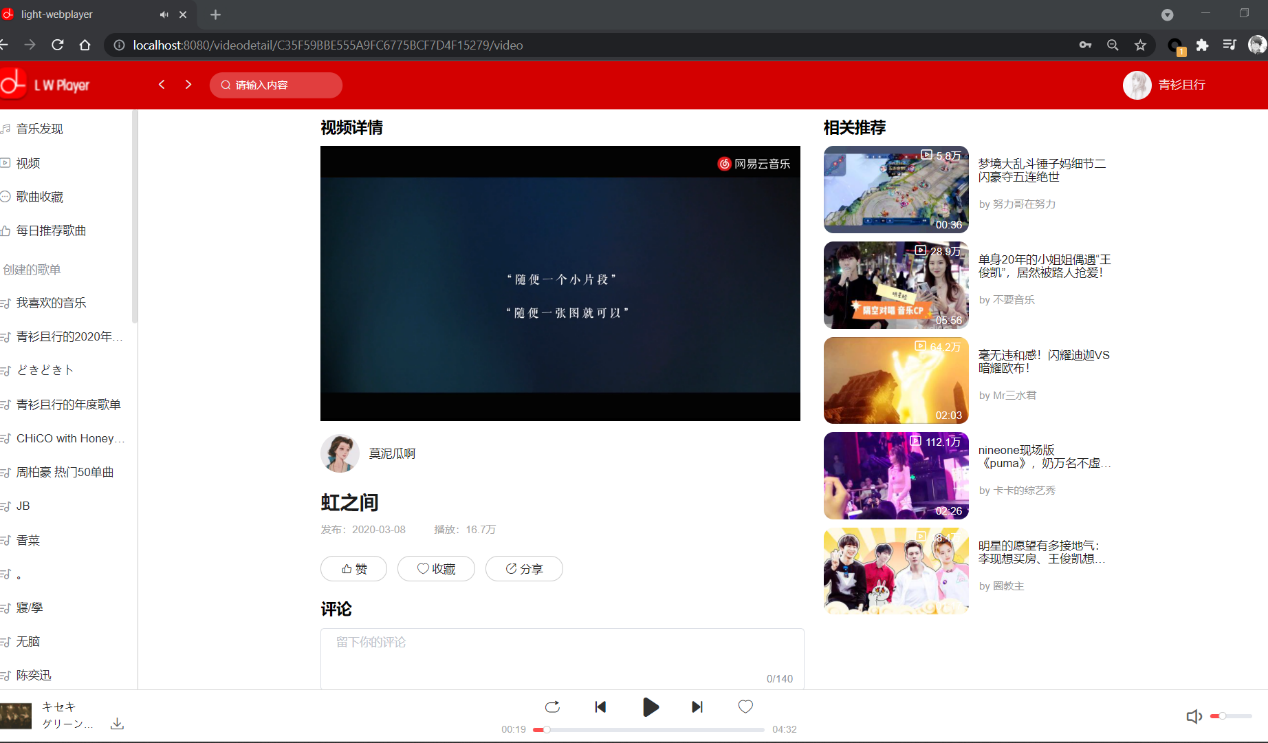
 点击左下角歌曲封面，可以实现歌词与评论的展示，如下图：

可以看到，歌词以及歌手信息、评论信息均能看到。下拉滚动条还能看到更多评论：

 点击下侧的发表评论按钮，可以进入评论发表页：

### 5.3.6 视频/mv播放效果测试

视频推荐页面：

视频/mv播放界面：

# 六、团队成员在项目中的角色与贡献

|  |  |  |
| --- | --- | --- |
| **姓名** | **角色** | **可验证贡献** |
| 陈柏钧 | 组长 | UI设计，后端网易云api接口服务参数记录和参数设计，logo以及ui图片收集，前端逻辑设计。 |
| 陈昊阳 | 前端开发 | 前端项目中，component文件下的组件开发 |
| 陈惟逸 | 前端开发  后端开发 | 前端项目所有代码（除去除component组件）的开发和所有代码的管理。  后端项目中接口的设计。 |
| 郑宗汕 | 后端开发 | 后端项目的所有函数开发。 |
| 冯珞轩 | 用户  文档管理 | 编写设计文档，作为客户提出需求以及与测试人员对接。 |
| 张钦博 | 用户  测试人员  文档管理 | 编写设计文档，作为用户与测试人员参与系统测试。 |
| 文柯力 | 测试人员  文档管理 | 编写测试文档，作为测试人员参与系统用户，与用户对接。 |

**教师评语评分**

评语：

评分：

评阅人：